Deploying the First PSTN-Based IoT Mechanism

Yi-Bing Lin, Chunghwa Telecom and National Chiao Tung University
Rong-Syh Lin, Yuan-Kai Chen, and Chin-Ywu Twu, Chunghwa Telecom
Shun-Ren Yang, National Tsing Hua University

Abstract

Existing IoT services are based on data communications technologies that do not involve the public switched telephone network (PSTN). Since the telephone numbers have been assigned to machine-type devices, PSTN switches can play a role in IoT service routing. In this article we deploy a PSTN-based IoT mechanism where the interaction between the users and the IoT devices is achieved through PSTN switches. To our knowledge, this is the first PSTN-based IoT solution in the world. With this mechanism, all PSTN customer premises equipment (CPE; fixed-line and mobile phones) can access IoT services without installing any software (mobile apps). By reusing the existing PSTN infrastructure, PSTN-based IoT offers telecom-grade service, security, and network management for IoT, which are very expensive to build in non-PSTN-based IoT. Our approach conveniently enables the existing CPE to access IoT applications, which will significantly promote the IoT service industry.

Introduction

Since Alexander Graham Bell (1847–1922) established the Bell Telephone Company in 1877, the public switched telephone network (PSTN) has been completely deployed worldwide, and telephone service has become one of the most successful businesses in history. In recent years, the Internet (data communications) is replacing PSTN (telecommunications), and the commercial PSTN voice service is being replaced by free voice services offered by over-the-top (OTT) services (Skype, LINE, WeChat, etc.). The revenue generated by the PSTN drops every year. On the other hand, the cost of maintaining the PSTN infrastructure remains high. Therefore, it is essential to reuse the PSTN for new applications. One promising direction is to integrate Internet of Things (IoT) services with the PSTN. This article reports on the deployment of a PSTN-based IoT mechanism. To our knowledge, this mechanism is the first PSTN-based IoT solution in the world. We use the term customer premises equipment (CPE) to represent a fixed-line telephone set (typically a touch-tone phone) or a mobile phone set (either a smartphone or a feature phone).

Today, 99 percent of IoT services are developed based on Internet technologies, and only some of them “slightly” utilize the telecom services, specifically, short messages to deliver alert information. IoT applications are accessed through cellular data service or non-proprietary wireless access environments such as public free Wi-Fi access points (APs). Users typically access the services through a web browser or an app installed on smartphones. Four important issues of Internet-based IoT can be effectively addressed by the PSTN.

First, the data path between the Internet-based IoT server and the smartphones may involve the public network (e.g., free Wi-Fi), which cannot guarantee the quality of service (QoS) and may cause security problems. In the PSTN, the data path between the CPE and the switch is established in the managed network (private network), which guarantees telecom-grade quality and security. The telephone number for an IoT device is the key to PSTN security. Since this number is given by the telephone company and is routed by the switch, no fraudulent usage may occur. The calling party can be validated through the traditional PSTN approaches. On the other hand, in the Internet-based IoT, the user may accidentally download unsecured mobile apps and be directed to fake application servers.

Second, In Internet-based IoT the users must install apps on their smartphones before they can access services. These apps consume significant amounts of power if the users put them in the foreground, and the smartphones may need to be recharged frequently. On the other hand, if the users put the apps in the background to save power, they may miss real-time events. In the PSTN, the CPE is designed for handling real-time events with low power consumption, and the IoT applications can be accessed through the telephone number.

Third, billing for Internet-based IoT services is typically simple and does not provide an intelligent charging mechanism. The billing mechanism for the PSTN has been developed for over 100 years, and is reliable and flexible. All IoT services can be effectively accommodated in the existing telecom billing mechanism, and are charged with the telephone bills without extra overhead [1].

Fourth, the cost for managing and maintaining an end-to-end Internet-based IoT platform with telecom-grade quality is very high. Telephone companies have developed telecom operations support systems (TOSSs) for intelligent service provisioning, integrated large-scale network management, trouble handling and resolution, and end-to-end service quality management [2]. It is not cost effective for any Internet-based IoT service provider to develop large-scale systems such as TOSSs. On the other hand, the telephone company can easily accommodate the IoT platform in existing TOSSs.

Assignment of telephone numbers to machine-type devices is an important mechanism that glues telephony with IoT. In Taiwan, 040 telephone number blocks are assigned to NB-IoT services [3]. Therefore, PSTN switches play an important role in routing the calls for the NB-IoT devices. The article is organized as follows. We describe the PSTN-based IoT mechanism. We show how PSTN CPE can provide the mobile app features. We elaborate on telecom-grade network and service maintenance for IoT.

PSTN-Based IoT Mechanism

In this section we first introduce the PSTN call setup procedure, and then show how this procedure can be used to interact with the IoT devices.

Figure 1 shows the simplified PSTN-to-mobile call setup [1] with the following steps.

Step A.1: The calling party (Fig. 1, 1) dials the mobile station ISDN number (MSISDN; i.e., the mobile phone number) of the called party (Fig. 1, 7). The Signal System Number 7 (SS7) Initial Address Message (IAM) is routed from the originating switch (Fig. 1, 2) to the gateway mobile switching center (GMSC; Fig. 1, 3) of the called party.

Step A.2: The GMSC queries the home subscriber server (HSS; Fig.1, 4) to obtain the mobile station roaming number (MSRN), the address of the terminating MSC (Fig. 1, 5) where the called party resides.

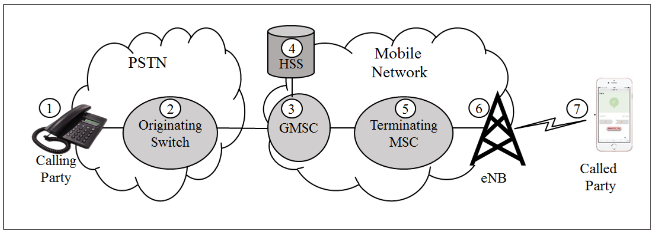

FIGURE 1. Simplified PSTN-to-mobile call setup.

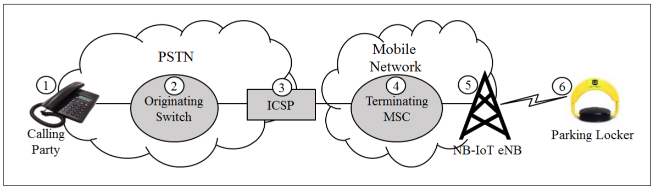

FIGURE 2. PSTN-based IoT call setup.

Step A.3: Based on the MSRN, the IAM message is routed to the terminating MSC and the terminating base station system (eNB; Fig. 1, 6) pages the called party (Fig. 1, 7). If the called party is in the radio coverage of the eNB and acknowledges the paging, the eNB sends the page response signal to the terminating MSC. The terminating MSC returns the SS7 Address Complete Message (ACM) to the originating switch. The ACM indicates that the routing information required to complete the call has been received by the terminating switch. A ringing signal is sent to the called party, and a ringback signal is sent to the calling party.

Step A.4: When the called party accepts the call, an answer signal is sent to the terminating MSC. The terminating MSC sends the SS7 Answer Message (ANM) to the originating switch. The ringing and the ringback tones are removed, and the conversation starts.

The above basic call setup model depends heavily on switch-based translations and features to provide the call processing intelligence. At the originating switch, we can utilize the concept of an advanced intelligent network to control call processing intelligence through software. Chunghwa Telecom has built such “intelligent software,” called the Intelligent Communications Service Platform (ICSP). To integrate PSTN with IoT, the ICSP allows the user to use PSTN CPE to interact with IoT devices (sensors and actuators). To support PSTN-based IoT using NB-IoT [3], the basic call setup Steps A.1–A.4 are modified as follows.

Step B.1. When the calling party (Fig. 2, 1) dials the MSISDN for a NB-IoT device, the originating switch (Fig. 2, 2) executes a global title translation (GTT) based on the dialed 040 number. The GTT of the switch directs the IAM message to the ICSP (Fig. 2, 3). In this step, the calling party may dial a service code instead of the 040 number (to be elaborated in the next section).

Step B.2. The ICSP suspends the original call signal. Based on the information provided in the IAM message, the ICSP continues to send a modified IAM message (where the calling party is replaced by the ICSP) to the NB-IoT device (e.g., a parking locker). If the service code is dialed in Step B.1, the code is translated into the corresponding 040 number in the modified IAM message.

Step B.3. This step is similar to Steps A.2 and A.3. Depending on how NB-IoT is implemented, the signaling paths to the NB-IoT devices may be different. The details can be found in [3].

After Step B.3, the ICSP sets up a call path back to the calling party (1–2–3, Fig. 2) and a data path (3–4–5–6) to the IoT device. This is a so-called back-to-back call. Note that at Step B.3, the ICSP may also route the calls to any sensors/actuators in the Internet domain instead of the mobile network. Interaction between the ICSP and the originating CPE is always completed through the SS7 signaling (IAM, ACM and ANM). To investigate the time complexity of the PSTN-base IoT approach, we have measured several delays defined below.

T1: the delay between when the originating switch sends the IAM message and when it receives the ACM message

T2: the delay between when the originating switch sends the IAM message and when it receives the ANM message

T3: the delay between when the ICSP sends the modified IAM message and when it receives the ACM message

T4: the delay between when the ICSP sends the modified IAM message and when it receives the ANM message

Figure 3 illustrates the histograms of the T1–T4 delays through over 2000 measurements, where the IoT device (Fig. 2, 6) controlled by the ICSP (Fig. 2, 3) is a dummy (i.e., the device responds to paging immediately without any delay). Therefore, we can identify the overheads of the PSTN-based IoT infrastructure.

Computed from the histograms, the average delays of T1 and T2 are 0.22 s and 0.89 s, respectively. The average delays of T3 and T4 are 0.013 s and 0.70 s, respectively. Note that T1–T3 and T2–T4 are roughly the same (about 0.2 s), and are the overheads of the interactions between the originating switch and the ICSP.

If the IoT device is a parking locker, we found that T2 is 30.03 s, and T2(locker)–T2(dummy) is roughly 29 s. This result indicates that the response of an actuator is likely to be the bottleneck of an execution where IoT devices are powered by batteries. To save energy, these IoT devices are exercised with sleep mode and therefore cannot respond immediately [4].

Replacing Mobile Apps by Telephone CPE

PSTN-based IoT does not need to install any software (i.e., mobile apps) in the CPE for accessing the IoT applications. Every IoT application is represented by a telephone number, and it is accessed by phone number dialing. There are already many ways to simplify the dialing process and assistance for users to memorize the phone numbers. The user can utilize speed dialing to quickly access the IoT applications. Speed dialing allows users to access frequently used numbers by dialing abbreviated codes even if the calling and called party lines are served by different switches. Speed dialing is available for almost all PSTN CPE. If the CPE is “smart enough” to provide an address book feature, the number of an IoT application can be included in the address book with an easy-to-memorize alias name. Our experience indicates that it takes the same time complexity or is easier to find a service among many phone numbers in an address book than flipping the touch screen of a smartphone to find the target service icon among the same number of icons.

We can also provide service codes where all users dial the same number to access different IoT devices. For example, all users dial the same xxx-427243 (xxx-garage) number to control their garage doors. This feature is provided by the caller ID mechanism. When this number is dialed, the ICSP (Fig. 2, 3) uses both the calling party’s phone number and xxx-427243 as the index to search the IoT phone number database to obtain the 040 number of the garage door controller of the user’s house and then set up the call following Step B.3. Service

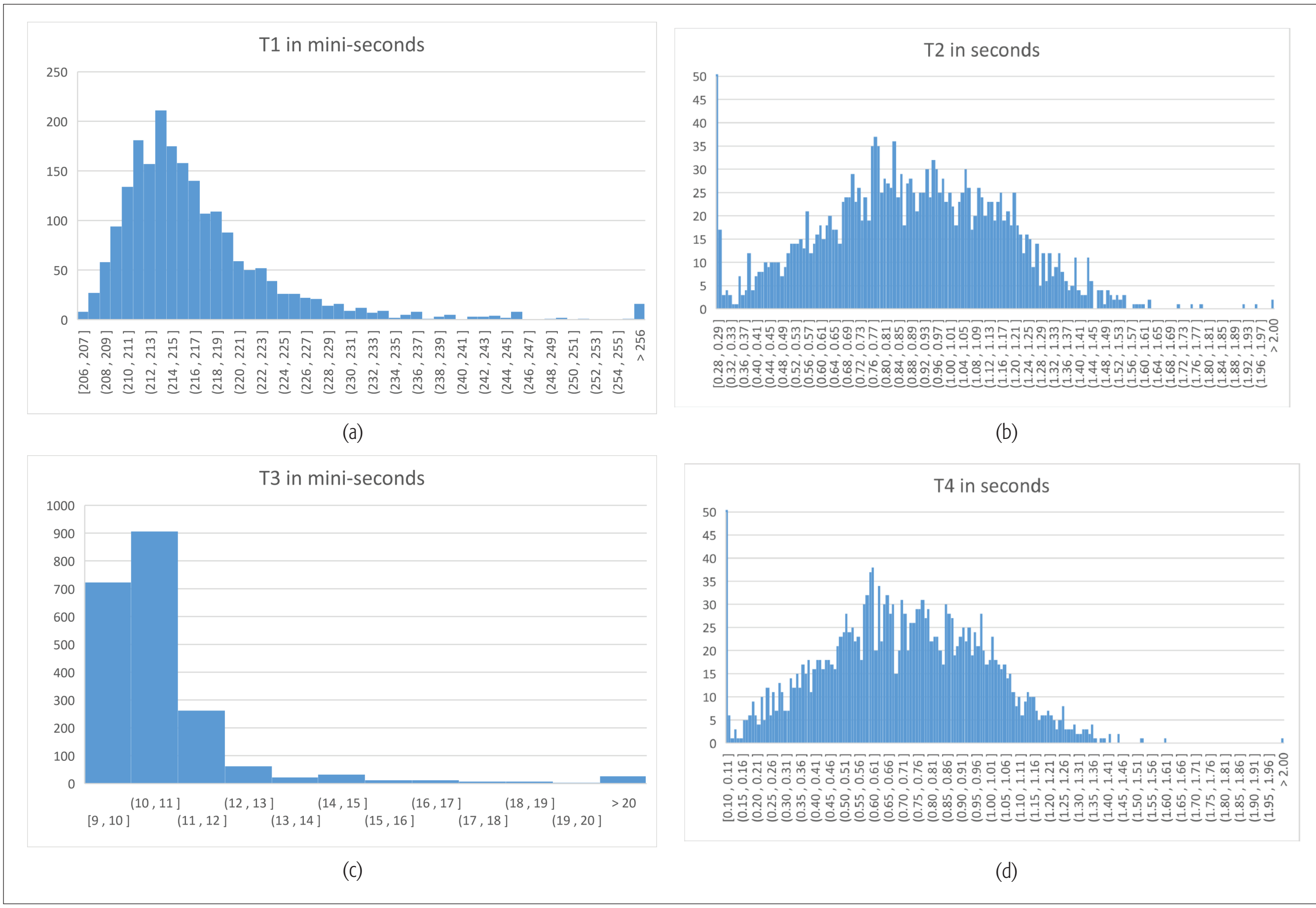

FIGURE 3. Histograms for T1, T2, T3, and T4.

codes can be used together with speed dialing/address book to simplify the dialing process for accessing the IoT applications. The user can also utilize one 040 number to access multiple IoT devices. This is achieved by the PSTN line extension feature where the 040 number is assigned to a private branch exchange (PBX), and the PBX's extension lines are connected to different IoT devices. The IoT PBX needs to interface the extension line to the IoT device control board (e.g., an Arduino board [5]), which may become a commercial product. This product serves as, for example, the smart home gateway to control home appliances.

After an IoT device is accessed, it can be controlled by the user through the well-known PSTN feature called interactive voice response (IVR). The ICSP is equipped with text-to-speech (TTS) and automatic speech recognition (ASR) mechanisms. Through TTS, the ICSP talks to the user to provide the IoT information. The user can give IoT instructions to the ICSP by pressing the keypad or just talk to the ICSP through ASR. If the user's CPE has the speaker phone feature, it can be operated just like Google Home or Amazon Alexa.

The user interacts with an IoT application in two scenarios. In the pull scenario, the user initiates the access to the IoT application. In PSTN-based IoT, the user dials the phone number of the IoT device directly. In the push scenario, the IoT device sends information to the user when the subscribed events occur. The push scenario is very energy consuming for Internet-based IoT, which requires putting the mobile app in the foreground. In PSTN-based IoT, the user is accessed by the IoT application by dialing his/her phone number. Waiting for an incoming call consumes little power at both fixed and mobile CPE (compared to mobile apps). In particular, the PSTN uses very little power to operate a telephone. The design of a fixed-line telephone set can always be powered with just a 9 V battery and a resistor, and will work as long as it is getting between 6 and 12 V at about 30 mA. The phone company supplies the power for a wired CPE through a dedicated copper pair (phone line). Even if the house loses electricity, the CPE is still powered through the phone line.

In Internet-based IoT, mobile apps typically use GPS for location services. It is well known that GPS is energy consuming. On the other hand, the telephone company knows the location of every fixed-line phone. Such information is very useful for some IoT services. For example, in Taiwan and Mexico, house holders need to keep their waste at their doors waiting for the waste collection vehicle to come to pick up the waste. The waste collectors indicate their readiness by ringing a distinctive bell and possibly shouting. The house holders may forget and miss the collection vehicle. With PSTN-based IoT, the house holders can dial the service code to subscribe to the waste collection alert service, which tells the ICSP to alert, say, 10 minutes before the collection vehicle arrives. The vehicle has a GPS-sensing NB-IoT device installed. From the location information of the waste collection vehicle and the location of the house (which is known through the telephone number of the house), the ICSP can alert the house holder by automatically dialing his/her phone number.

Business Support System/Operations Support System for IoT

With the well-developed business support system/operation support system (BSS/OSS), telephone companies can manage the complexity of service and network to converge voice, data, and video, which promptly and effectively provide competitive services upon customer request, and offer differentiated services that meet customer expectations.

In Chunghwa Telecom, a TOSS has been developed for the support of fulfillment and assurance operations of IoT services based on a suite of New generation operations systems and software (NGOSS) [2]. Four OSS groups, TOSS-P (service provisioning), TOSS-N (network management), TOSS-T (trouble resolution), and TOSS-Q (service quality management), are organized and integrated following the standard telecom operation processes. These NGOSS group functions are more than enough to support and maintain any IoT service platform. Specifically, end-to-end *fulfillment process flow* is achieved by TOSS-P and TOSS-N to promptly provide requested IoT service products. TOSS-T and TOSS-Q work with TOSS-N to support end-to-end *assurance process flow* to ensure that the IoT services are continuously available to meet the QoS performance levels. TOSS-P, TOSS-T, and TOSS-Q provide user interfaces for handling customer orders and trouble appeals. TOSS-N manages the multi-technology resources (e.g., IoT platforms and protocols [6–8]) and hides the resource management complexity from TOSS-P, TOSS-T, and TOSS-Q.

TOSS allows the designer to define error/exception handling in an IoT service provision sequence. For example, if the IoT system is disconnected or temporarily fails during the activation process, an exception-handling process is activated to resolve the problem and inform the operators as well. The designer sets the alert criteria to monitor the status of a provision sequence. For example, if the IoT service platform does not complete the provision task in 10 minutes, TOSS-P will report an exception alert to the system administrator.

TOSS-Q interacts with TOSS-N to provide testing functions for the physical layer, data link layer, and IP layer. The IoT service operation center can utilize these functions to carry out commercial IoT system tests such as service lock/unlock, service platform configuration query, and customer action query. These functions can also be used to test expensive IoT devices. For example, in a smart building application, TOSS-Q automatically conducts elevator system tests including elevator car reset, elevator firmware update, elevator status query, and so on. At National Chiao Tung University, we have installed inside an elevator car a 3D accelerometer, a barometer, and a thermometer inside an elevator. Outside the car, the motor is attached to a biaxial vibration accelerator. These sensor data can be used by TOSS-Q to monitor the statuses of elevator cars. If abnormal situations are detected, TOSS-T will generate trouble tickets, send them to the appropriate personnel (e.g., network maintenance operator, IoT service provider, or elevator manufacturer), and track these tickets until the services are restored.

Conclusions

This article reports on the deployment of the first PSTN-based IoT mechanism in the world, where the interaction between users and IoT devices is achieved through PSTN switches. With this mechanism, about 10 million PSTN CPEs in Chunghwa Telecom can be automatically enabled to access IoT services without installing any software (mobile apps). By reusing the existing PSTN infrastructure, PSTN-based IoT offers telecom-grade IoT service, security, and network management, which are very expensive to build in Internet-based IoT.

Acknowledgment

The authors would like to thank Shian-Ming Chen, Yikai Chiang, Gong-Da Fan, Chao-Chun Huang, Jen-Hong Ju, and Chung-Shih Tang of CHT Telecom Laboratories, and Ta-Sheng Chang, Jun-Ru Huang, and Zhi-Hong Lin of CHT Mobile for their valuable support. This work was supported in part by the Ministry of Education through the SPROUT Project-Center for Open Intelligent Connectivity of National Chiao Tung University, Taiwan.

References

[1] Y.-B. Lin and S.-I. Sou, *Charging for Mobile All-IP Telecommunications*, Wiley, 2008.
[2] Y.-K. Chen *et al.*, "TOSS: Telecom Operations Support Systems for Broadband Services," *J. Info. Processing Systems*, vol. 6, no. 1, 2010, pp. 1–20.
[3] 3GPP TR 23.720, "Study on Architecture Enhancements for Cellular Internet of Things", v13.0.0, Mar. 2016.
[4] S.-R. Yang and Y.-B. Lin, "Modeling UMTS Discontinuous Reception Mechanism," *IEEE Trans. Wireless Commun.*, vol. 4, no. 1, 2005.
[5] Y.-W. Lin *et al.*, "ArduTalk: An Arduino Network Application Development Platform Based on IoTtalk," *IEEE System J.*, 28 Nov. 2017, pp. 1–9.
[6] oneM2M: Standards for M2M and the Internet of Things; http://www.onem2m.org/ 384, accessed 22 May 2018.
[7] Open Connectivity Foundation; https://openconnectivity.org/, accessed 22 May 2018.
[8] ETSI. TS 102 921 — V1.3.1 — Machine-to-Machine Communications (M2M); MIA, DIA and MID Interfaces; http://www.etsi.org/deliver/etsi_ts/102900_102999/102921/01.03.01_60/ts_102921v010301p.pdf 387, accessed 22 May 2018.